Name:

ID:

CSCI 3104, Algorithms
Quiz 4A

Profs. Chen & Grochow
Spring 2020, CU-Boulder

Instructions: This quiz is closed book and an individual effort. Electronic devices are NOT allowed. Possession of such electronics is grounds to receive a 0 on this quiz. Proofs should be written in **complete sentences. Show and justify all work to receive full credit.**

Please provide these:
Left neighbor name :
Right neighbor name :

Name: [ ]

ID: [ ]

CSCI 3104, Algorithms **Profs. Chen & Grochow**

Quiz 4A **Spring 2020, CU-Boulder**

1. **Standard 9.** *Within the context of randomized QuickSort, (a) state the expected runtime (worst input, expected random behavior) and (b) the worst-case runtime (worst input, worst random behavior), and (c) explain what in the algorithm accounts for the difference between the two.*

Solution:

(a) *The worst input, expected runtime of QuickSort is* $\mathcal{O}(n\log(n))$

(b) *The worst input, worst-case runtime of QuickSort is* $\mathcal{O}(n^2)$

(c) *While the best case choice of pivot, the median, only occurs with probability* $\frac{1}{n}$*, in expectation we will select a pivot in the middle 50% of the data half of the time, which gives on average* $\mathcal{O}(n\log n)$ *overall runtime.*

In more detail:
We achieve the best-case runtime of QuickSort when the pivot is set to the median of the array A*, and the worst case runtime when the pivot is set at the max or min. Assuming that we choose the pivot uniformly at random, the median itself has only a* $\frac{1}{n}$ *chance of being chosen. However, we can consider instead the middle 50% of* A*, i.e. the portion of the array that falls between the lower and upper quartiles. In this case, the worst case runtime is* $T(n) = T(\frac{n}{4}) + T(\frac{3n}{4}) + \Theta(n) = \Theta(n\log_{\frac{4}{3}} n)$*. In expectation, we only need* **2** *recursive calls to get a pivot that lies between the quartiles. Thus, on average, the height of the recurrence tree will be some constant multiple of* $\log n$*, giving us an expected runtime bound of* $\mathcal{O}(n\log n)$

Name:

ID:

CSCI 3104, Algorithms | **Profs. Chen & Grochow**

Quiz 4A | **Spring 2020, CU-Boulder**

2. ***Standard 10.*** *Hash tables and balanced binary trees can be both be used to implement a dictionary data structure, which supports* `Add`*,* `Lookup`*, and* `Remove` *operations. In balanced binary trees containing n elements, the runtime of all operations is* $\Theta(\log n)$*.*

   For each of the following three scenarios, compare the performance of a dictionary implemented with a hash table (which resolves collisions with chaining using linked lists) to a dictionary implemented with a balanced binary tree.

   Parts (b) and (c) on next page.

   (a) *A hash table with hash function* $h_1(x) = 1$ *for all keys* x*.*

   Solution:

   The key is to identify the expected collision length since `Lookup` *and* `Remove` *are of the same complexity with expected collision length. Please note that we are using linked list, so* `Add` *is constant, e.g.,* $\Theta(1)$ *for all hash functions.*

   With $h_1(x) = 1$*, the expected collision length is* $\frac{n}{1}$*. So* `Lookup` *and* `Remove` *are both with* $\Theta(n)$*.*